1. Risk of a Software Crisis

Return to Question 11

For these reasons, the Object Management Group (OMG) recommends the adoption of a systematic effort for the development of an SoS identified as a Mission-Critical SoS. The CBDC also has a potential System Lifcycle that spans decades at a minimum. **The need for an SoS, long-lived, Mission-Critical System sets the stage for the biggest risks for the U.S. CBDC**, the potential for a looming Software Crisis.

A **Software Crisis** occurs on projects for many reasons, but the Information Technology (IT) industry has focused on a shortlist, which is provided in summary form in Table 1. Any particular project suffering a **Software Crisis** may have any number of these issues and unfortunately, some projects might have all of these issues.

Table 1: Issues causing a Project to have a **Software Crisis**.

1. Projects are running over budget
2. Projects are running behind schedule
3. Poor quality of the delivered software
4. Poor definition of requirements
5. Poor adherence to the requirements
6. Poor management of the entire project throughout its lifecycle
7. Poor communications between the Stakeholders, Systems Engineers, and Software Engineers
8. Poor documentation of Policies and Procedures for the project
9. Poor enforcement of Policies and Procedures for the project
10. Poor training of Stakeholders, Systems Engineers, and Software Engineers on Policies and Procedures.
11. Increase in System and Software complexity
12. Increase in Software costs compared to Hardware

However, all is not lost for the CBDC. There is a way to prevent a future CNDC **Software Crisis** by applying sound Systems Engineering practices throughout the CBDC lifecycle, starting immediately. The OMG has a rich history of working in Systems and Software Engineering. Table 2 provides a list of OMG standards covering Systems Engineering and Software Engineering.

Table 2: The Object Management Group's list of System and Software Engineering Stadards.

- Business Motivation Model (BMM)
- Business Process Model and Notation (BPMN)
- Common Warehouse Metamodel (CWM)
- Distributed Ontology, Model, and Specification Language (DOL)
- Financial Industry Business Ontology (FIBO)
- Financial Instrument Global Identifier (FIGI)
- MetaObject Facility (MOF)
- Model Based Systems Engineering (MBSE)
- Model Driven Architecture (MDA)

- Ontology Definition Metamodel (ODM)
- Semantics Of Business Vocabulary And Business Rules (SBVR)
- Structured Assurance Case Metamodel (SACM)
- Systems Modeling Language (SysML)
- Unified Architecture Framework (UAF)
- Unified Modeling Language (UML)
- XML Metadata Interchange (XMI)
- Systems and software engineering -- System life cycle processes
- Measurement of System and Software Product Quality

The Systems Engineering process as described by the Department of Energy (DOE) is to develop, manage, and implement large programs [1]. The following is a modified version of the DOE process tweaked to differentiate the Water Fall Model versus the Agile Model.

1. Orderly definition of the **System** through top-down development of **System Functions** and **System Requirements**. This is an iterative process with each iteration providing further decomposition of the System Level Requirments as needed. **Note:** These Systems-level definition iterations should not be confused with the Agile Sprints used during development. See the OMG DIDO-RA section on The Current State of DIDO Requirements.
2. Clear distinction between: a. External driven **System** requirements and constraints, which are by intent not easy to modify. In other words, the identification of System Functional and Non-Functional Requirements. b. Internal driven **Design** (i.e., implementation) requirements developed by the project, which are potentially modifiable and evolutionary with new requirements added as the system is developed. In other words, Derived Requirements.
3. Top-down consideration and evaluation of alternative solutions and designs based on the System Functional and Non-Functional Requirements
4. Completeness and traceability for the design of System Components and System Interfaces, for configuration and change control, and for the system verification and validation plan(s). In other words, the SoS must come together as a cohesive, solitary group of capabilities that synergize the system to deliver the desired effects. All the Systems within the SoS must have a single, cohesive, unified understanding of the other Systems within the SoS and must be able to use standardized Application Programming Interfaces (APIs).

DOE goes on to describe the value of the Systems Engineering Process realization in a number of ways, including:

1. Increased ability to estimate system life-cycle costs
2. Reduced redesign due to consideration of the entire system throughout its development
3. Increased ability to affect design changes and retrofits due to clear traceability of requirements, design features, and configuration control
4. Increased probability of achieving the best technical design and operational concept through the iterative consideration of design alternatives, where ***best*** is defined through decision criteria such as cost, risk, and use. See the OMG DIDO-RA section on Assessing Requirements

Figure 1 provides a simplified high-level processes flow for Systems Engineering. This process was developed by the U.S. Department of Energy and would have to be tailored to meet the needs of a U.S. CBDC. Basically, the System's process flow is captured in Table ##REF:sysProcSteps##

Figure 1: Simplified high-level Systems Engineering Process as defined by the U.S. Department of Energy.

The steps in the Simplified high-level Systems Engineering Process as defined by the U.S. Department of Energy:

1. A high-level statement of system needs. In this discussion, the Mission needs are referred to as “Desirements”. The current “desirements” from the White Paper as identified by the Object Management Group's report called White Paper Analysis is a good starting point for these.

2. The *“Mission Needs”* are analyzed and transformed into *“Mission Statements”* (i.e., Systems Requirements). For example, the **White Paper** desirement of:

The Federal Reserve does not intend to proceed with the issuance of a CBDC without clear support from the Executive Branch, the Legislative Branch, and also ideally in the form of a specific authorizing law would be transformed into:

```
* U.S. CBDC shall be authorized by a specific U.S. Law
* U.S. CBDC Authorizing Law shall be approved by the Legislative Branch
* U.S. CBDC Authorizing Law shall be approved by the Executive Branch
```

3. The *“Mission Statements”* are transformed into Functional (i.e, performance, interfaces)and Non-Functional Requirements (i.e., constraints), see OMG DIDO-RA Specifying Requirements. Also see the OMG DIDO-RA section on Testability and especially the subsection on Software Assurance (SwA). If the requirements are not testable, then they serve little purpose.

4. The *“Requirements”* are allocated to Systems, or components (i.e., elements) and added to a formal System Description and Systems Analysis. Table 3 captures the documents called out in the Unified Architecture Framework (UAF). These documents can be tailored for the U.S. CBDC, but many of the documents are useful for Mission Critical Systems.

Table 3: The kinds of documents that can be used to define the system.

Viewpoint	Acronym	Description
Architecture Management	**Am**	Identifies the metadata and views required to develop a suitable architecture that is fit for its purpose.
Strategic	**St**	Capability management process. Describes the capability taxonomy, composition, dependencies, and evolution.
Operational	**Op**	Illustrates the Logical Architecture of the enterprise. Describes the requirements, operational behavior, structure, and exchanges required to support (exhibit) capabilities. Defines all operational elements in an implementation/solution-independent manner.

Viewpoint	Acronym	Description
Services	**Sv**	The Service-Orientated View (SOV) is a description of services needed to directly support the operational domain as described in the Operational View. A service within MODAF is understood in its broadest sense, as a unit of work through which a provider provides a useful result to a consumer. MODAF: The Service Views within the Services Viewpoint describe the design for service-based solutions to support operational development processes (JCIDS) and Defense Acquisition System or capability development within the Joint Capability Areas.
Personnel	**Ps**	Defines and explores organizational resource types. Shows the taxonomy of types of organizational resources as well as connections, interaction, and growth over time.
Resources	**Rs**	Captures a solution architecture consisting of resources, e.g., organizational, software, artifacts, capability configurations, and natural resources that implement the operational requirements. Further design of a resource is typically detailed in SysML or UML.
Security	**Sc**	Security assets and security enclaves. Defines the hierarchy of security assets and asset owners, security constraints (policy, laws, and guidance), and details where they are located (security enclaves).
Projects	**Pj**	Describes projects and project milestones, how those projects deliver capabilities, the organizations contributing to the projects, and dependencies between projects.
Standards	**Sd**	MODAF: Technical Standards Views are extended from the core DoDAF views to include non-technical standards such as operational doctrine, industry process standards, etc. DoDAF: The Standards Views within the Standards Viewpoint are the set of rules governing the arrangement, interaction, and interdependence of solution parts or elements.
Actual Resources	**Ar**	The analysis, e.g., evaluation of different alternatives, what-if, trade-offs, V&V on the actual resource configurations. Illustrates the expected or achieved actual resource configurations.
Motivation	**Mv**	Captures motivational elements e.g., challenges, opportunities, and concerns, that pertain to enterprise transformation efforts, and different types of requirements, e.g., operational, services, personnel, resources, or security controls.
Taxonomy	**Tx**	Presents all the elements as a standalone structure. Presents all the elements as a specialization hierarchy, provides a text definition for each one and references the source of the element
Structure	**Sr**	Describes the breakdown of structural elements e.g., logical performers, systems, projects, etc. into their smaller parts
Connectivity	**Cn**	Describes the connections, relationships, and interactions between the different elements.
Processes	**Pr**	Captures activity-based behavior and flows. It describes activities, their Inputs/Outputs, activity actions, and flows between them.
States	**St**	Captures state-based behavior of an element. It is a graphical representation of the states of a structural element and how it responds to various events and actions.

Viewpoint	Acronym	Description
Sequences	**Sq**	Expresses a time-ordered examination of the exchanges as a result of a particular scenario. Provides a time-ordered examination of the exchanges between participating elements as a result of a particular scenario.
Information	**If**	Address the information perspective on operational, service, and resource architectures. Allows analysis of an architecture's information and data definition aspect, without consideration of implementation-specific issues.
Constraints	**Ct**	Details the measurements that set performance requirements constraining capabilities. Also defines the rules governing behavior and structure.
Roadmap	**Rm**	Addresses how elements in the architecture change over time.
Traceability	**Tr**	Describes the mapping between elements in the architecture. This can be between different viewpoints within domains as well as between domains. It can also be between structure and behaviors.

5. Verification of the System is progressing according to plan. This is done through Test Inspection, Demonstrations, as well as Static and Dynamic Analysis of the System. Another major tool should be the use of Modeling and testing in Virtual Environments.

6. Evaluation and Optimization occur before a release to the public. Based on the results of Trade Studies, risk analysis, performance, etc, the System Design Specifications can be updated, refined, or added to. </WRAP>

[1] National Academy of Sciences, Systems Analysis and Systems Engineering in Environmental Remediation Programs at the Department of Energy Hanford Site, National Research Council 1998. Systems Analysis and Systems Engineering in Environmental Remediation Programs at the Department of Energy Hanford Site. Washington, DC: The National Academies Press. https://doi.org/10.17226/6224